Sua rede Wireless é Realmente Segura?

Um dos grandes problemas numa redes wireless é que os sinais são transmitidos pelo ar. Os pontos de acesso e placas utilizam por padrão antenas baratas, que proporcionam um alcance reduzido, fazendo com que o sinal da sua rede possa ser capturado de muito mais longe por alguém com uma antena de alto ganho.

Não existe como impedir que o sinal se propague livremente pelas redondezas (a menos que você pretenda ir morar num bunker, com paredes reforçadas com placas de aço), de forma que a única forma eficaz de proteção é encriptar toda a transmissão, fazendo com que as informações capturadas não tenham serventia.

Como esta questão da segurança em redes wireless é muito divulgada, a maior parte das redes já utiliza algum tipo de proteção, seja ativando o WEP ou WPA, seja bloqueando os micros que podem se conectar ao ponto de acesso com base no endereço MAC.

Este tópico se destina a mostrar como é fácil burlar a maioria destas proteções e quebrar a encriptação do WEP (inclusive do WEP de 128 bits), usando ferramentas simples.

É melhor que você conheça os ataques mais usados e veja você mesmo como é possível derrubar cada uma das proteções que utilizamos numa rede típica, do que ficar com um falso senso de segurança, achando que o WEP de 128 é inquebrável, que não é possível detectar um ponto de acesso com o SSID broadcast desativado, ou que não é possível burlar a restrição de acesso baseada no endereço MAC usada em muitas redes.

Usando o Kismet

O Kismet é uma ferramenta poderosa, que pode ser usado tanto para checar a segurança de sua própria rede wireless, quanto para checar a presença de outras redes próximas e assim descobrir os canais que estão mais congestionados (configurando sua rede para usar um que esteja livre) ou até mesmo invadir redes. O Kismet em sí não impõe restrições ao que você pode fazer. Assim como qualquer outra ferramenta, ele pode ser usado de forma produtiva ou destrutiva, de acordo com a índole de quem usa.

A página do projeto é a: http://www.kismetwireless.net/.

A principal característica do Kismet é que ele é uma ferramenta passiva. Ao ser ativado, ele coloca a placa wireless em modo de monitoramento (rfmon) e passa a escutar todos os sinais que cheguem até sua antena. Mesmo pontos de acesso configurados para não divulgar o ESSID ou com a encriptação ativa são detectados.

Como ele não transmite pacotes, apenas escuta as transmissões, todo o processo é feito sem prejudicar as redes vizinhas e de forma praticamente indetectável. A principal limitação é que, enquanto está em modo de monitoramento, a placa não pode

ser usada para outros fins. Para conectar-se a uma rede, você precisa primeiro parar a varredura.

Esta questão da detecção dos pontos de acesso com o ESSID desativado é interessante. Não é possível detectá-los diretamente, pois eles não respondem a pacotes de broadcast (por isso eles não são detectados por programas como o Netstumbler), mas o Kismet é capaz de detectá-los quando um cliente qualquer se associa a eles, pois o ESSID da rede é transmitido de forma não encriptada durante o processo de associação do cliente.

A partir daí, o Kismet passa a capturar todos os pacotes transmitidos. Caso a rede esteja encriptada, é possível descobrir a chave de encriptação usando o aircrack (que veremos a seguir), permitindo tanto escutar as conexões, quanto ingressar na rede.

Como o Kismet é uma das ferramentas mais usadas pelos crackers, é sempre interessante usá-lo para verificar a segurança da sua própria rede. Tente agir como algum vizinho obstinado agiria, capturando os pacotes ao longo de alguns dias. Verifique a distância de onde consegue pegar o sinal de sua rede e quais informações consegue descobrir. Depois, procure meios de reforçar a segurança da rede e anular o ataque.

Por ser uma ferramenta popular, ele está disponível na maioria as distribuições. Algumas, como o Knoppix (a partir da versão 3.7), já o trazem instalado por padrão.

Nas distribuições derivadas do Debian, você pode instalá-lo via apt-get:

 # apt-get install kismet

Antes de ser usar, é preciso configurar o arquivo "**/etc/kismet/kismet.conf**", especificando a placa wireless e o driver usado por ela, substituindo a linha:

source=none,none,addme

Por algo como:

source=madwifi_ag,ath0,atheros

... onde o "madwifi_ag" é o driver usado pela placa (que você pode verificar usando o comando **lspci**). Na documentação do Kismet o driver é chamado de "capture source", pois é a partir dele que o Kismet obtém os pacotes recebidos.

o "ath0" é a interface (que você vê através do comando **ifconfig**) e o "atheros" é um apelido para a placa (que você escolhe), com o qual ela será identificada dentro da tela de varredura.

Isto é necessário, pois o Kismet precisa de acesso de baixo nível ao hardware. Isto faz com que a compatibilidade esteja longe de ser perfeita. Diversas placas não funcionam em conjunto com o Kismet, com destaque para as placas que não possuem drivers nativos e precisam ser configurados através do ndiswrapper. Se você pretende usar o Kismet, o ideal é pesquisar antes de comprar a placa. Naturalmente, para que possa ser usada no Kismet, a placa precisa ter sido detectada pelo sistema, com a ativação dos módulos de Kernel necessários. Por isso, prefira sempre usar uma distribuição recente, que traga um conjunto atualizado de drivers. O Kurumin e o Kanotix estão

entre os melhores neste caso, pois trazem muitos drivers que não vem pré instalados em muitas distribuições.

Você pode ver uma lista detalhada dos drivers de placas wireless disponíveis e como instalar manualmente cada um deles no meu livro *Linux Ferramentas Técnicas*.

Veja uma pequena lista dos drivers e placas suportados no Kismet 2006-04-R1:

acx100: O chipset ACX100 foi utilizado em placas de diversos fabricantes, entre eles a DLink, sendo depois substituído pelo ACX111. O ACX100 original é bem suportado pelo Kismet, o problema é que ele trabalha a 11 megabits, de forma que não é possível testar redes 802.11g.

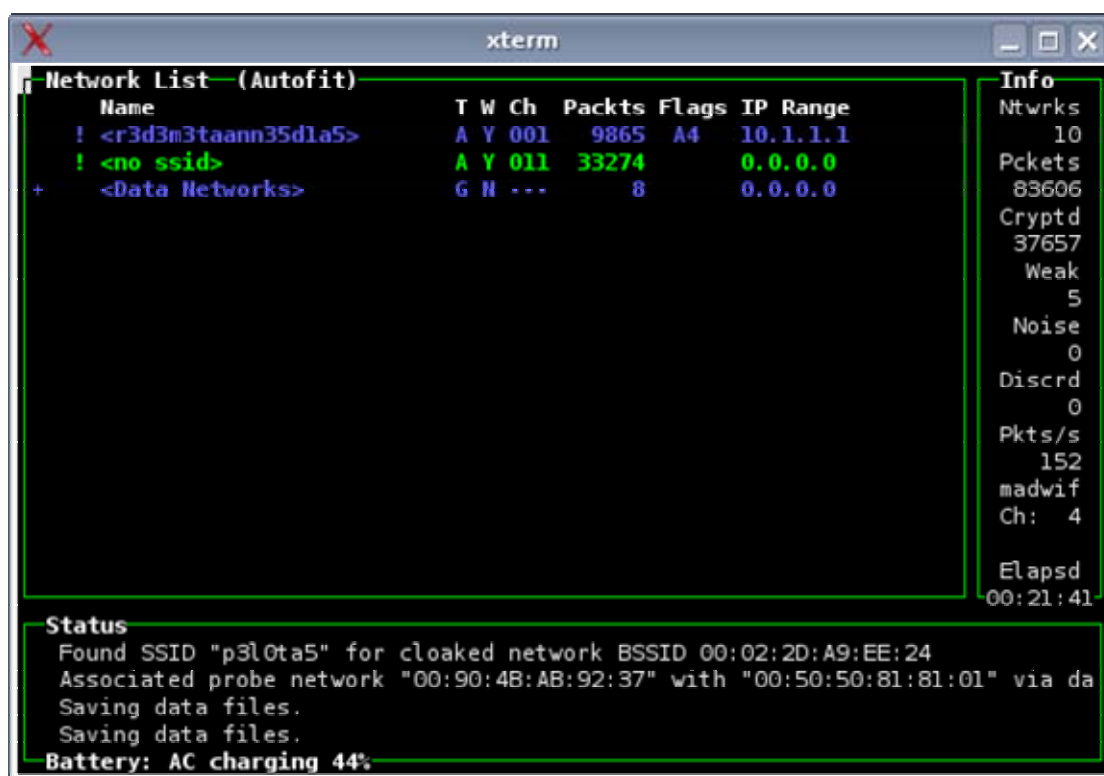
admtek: O ADM8211 é um chipset de baixo custo, encontrado em muitas placas baratas. Ele é suportado no Kismet, mas possui alguns problemas. O principal é que ele envia pacotes de broadcast quando em modo monitor, fazendo com que sua varredura seja detectável em toda a área de alcance do sinal. Qualquer administrador esperto vai perceber que você está capturando pacotes.

bcm43xx: As placas com chipset Broadcom podiam até recentemente ser usadas apenas no ndiswrapper. Recentemente, surgiu um driver nativo (http://bcm43xx.berlios.de) que passou a ser suportado no Kismet. O driver vem incluído por padrão a partir do Kernel 2.6.17, mas a compatibilidade no Kismet ainda está em estágio experimental.

ipw2100, **ipw2200**, **ipw2915** e **ipw3945**: Estes são os drivers para as placas Intel, encontrados nos notebooks Intel Centrino. O Kismet suporta toda a turma, mas você precisa indicar o driver correto para a sua placa entre os quatro.
O ipw2000 é o chipset mais antigo, que opera a 11 megabits; o ipw2200 é a segunda versão, que suporta tanto o 8011.b, quanto o 802.11g; o ipw2915 é quase idêntico ao ipw2200, mas suporta também o 802.11a, enquanto o ipw3945 é uma versão atualizada, que é encontrada nos notebooks com processadores Core Solo e Core Duo.

madwifi_a, **madwifi_b**, **madwifi_g**, **madwifi_ab** e **madwifi_ag**: Estes drivers representam diferentes modos de operação suportados pelo driver madwifi (http://sourceforge.net/projects/madwifi/), usado nas placas com chipset Atheros. Suportam tanto o driver madwifi antigo, quanto o madwifi-ng.
Usando os drivers madwifi_a, madwifi_b ou madwifi_g, a placa captura pacotes apenas dentro do padrão selecionado (o madwifi_a captura apenas pacotes de redes 802.11a, e assim por diante). O madwifi_g é o mais usado, pois captura simultaneamente os pacotes de redes 802.11b e 802.11g. O madwifi_ag, por sua vez, chaveia entre os modos "a" e "g", permitido capturar pacotes de redes que operam em qualquer um dos três padrões, mas num ritmo mais lento, devido ao chaveamento.

rt2400 e **rt2500**: Estes dois drivers dão suporte às placas com chipset Ralink, outro exemplo de chipset de baixo custo que está se tornando bastante comum. Apesar de não serem exatamente "placas de alta qualidade", as Ralink possuem um bom suporte no Linux, graças em parte aos esforços do próprio fabricante, que abriu as especificações e fornece placas de teste para os desenvolvedores. Isto contrasta com a atitude hostil de alguns fabricantes, como a Broadcom e a Texas (que fabrica os chipsets ACX).

rt8180: Este é o driver que oferece suporte às placas Realtek 8180. Muita gente usa estas placas em conjunto com o ndiswrapper, mas elas possuem um driver nativo,

disponível no http://rtl8180-sa2400.sourceforge.net/. Naturalmente, o Kismet só funciona caso seja usado o driver nativo.

prism54g: Este driver dá suporte às placas com o chipset Prism54, encontradas tanto em versão PCI ou PCMCIA, quanto em versão USB. Estas placas são caras e por isso relativamente incomuns no Brasil, mas são muito procuradas entre os grupos que fazem wardriving, pois as placas PCMCIA são geralmente de boa qualidade e quase sempre possuem conectores para antenas externas, um pré-requisito para usar uma antena de alto ganho e assim conseguir detectar redes distantes.

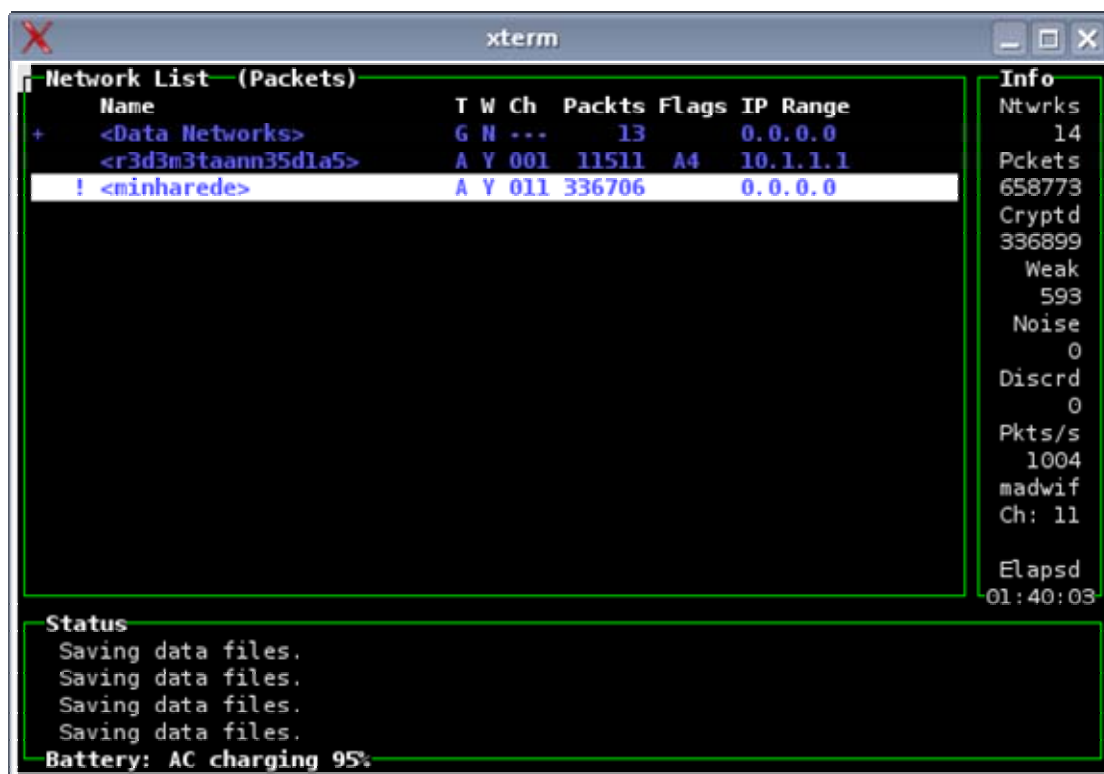
orinoco: Os drivers para as placas com chipset Orinoco (como as antigas Orinoco Gold e Orinoco Silver) precisam de um conjunto de patches para funcionar em conjunto com o Kismet, por isso acabam não sendo placas recomendáveis. Você pode ver detalhes sobre a instalação dos patches no http://www.kismetwireless.net/HOWTO-26_Orinoco_Rfmon.txt.

Depois de definir o driver, a interface e o nome no "**/etc/kismet/kismet.conf**", você pode abrir o Kismet chamando-o como root:

 # kismet

Inicialmente, o Kismet mostra as redes sem uma ordem definida, atualizando a lista conforma vai descobrindo novas informações. Pressione a tecla "**s**" para abrir o menu de organização, onde você pode definir a forma como a lista é organizada, de acordo com a qualidade do canal, volume de dados capturados, nome, etc. Uma opção comum (dentro do menu sort) é a "c", que organiza a lista baseado no canal usado por cada rede.

Por padrão, o Kismet chaveia entre todos os canais, tentando detectar todas as redes disponíveis. Neste modo, ele captura apenas uma pequena parte do tráfego de cada rede, assim como você só assiste parte de cada programa ao ficar zapiando entre vários canais da TV.

Selecione a rede que quer testar usando as setas e pressione "**shift + L**" (**L** maiúsculo) para travá-lo no canal da rede especificada. A partir daí ele passa a concentrar a atenção numa única rede, capturando todos os pacotes transmitidos:

Você pode também ver informações detalhadas sobre cada rede selecionando-a na lista e pressionando **enter**. Pressione "**q**" para sair do menu de detalhes e voltar à tela principal.

Outro recurso interessante é que o Kismet avisa sobre "clientes suspeitos", micros que enviam pacotes de conexão para os pontos de acesso, mas nunca se conectam a nenhuma rede, indício de que provavelmente são pessoas fazendo wardriving ou tentando invadir redes. Este é o comportamento de programas como o Netstumbler (do Windows). Micros rodando o Kismet não disparam este alerta, pois fazem o scan de forma passiva:

> ALERT: Suspicious client 00:12:F0:99:71:D1 - probing networks but never participating.

O Kismet gera um dump contendo todos os pacotes capturados, que vai por padrão para a pasta "**/var/log/kismet/**". A idéia é que você possa examinar o tráfego capturado posteriormente usando o Ethereal. O problema é que, ao sniffar uma rede movimentada, o dump pode se transformar rapidamente num arquivo com vários GB, exibindo que você reserve bastante espaço no HD.

Um dos maiores perigos numa rede wireless é que qualquer pessoa pode capturar o tráfego da sua rede e depois examiná-lo calmamente em busca de senhas e outros dados confidenciais transmitidos de forma não encriptada. O uso do WEP ou outro sistema de encriptação minimiza este risco, pois antes de chegar aos dados, é necessário quebrar a encriptação.

Evite usar chaves WEP de 64 bits, pois ele pode ser quebrado via força bruta caso seja possível capturar uma quantidade razoável de pacotes da rede. As chaves de 128 bits são um pouco mais seguras, embora também estejam longe de ser inquebráveis. Em termos se segurança, o WPA está à frente, mas usá-lo traz problemas de compatibilidade com algumas placas e drivers.

Sempre que possível, use o SSH, SSL ou outro sistema de encriptação na hora de acessar outras máquinas da rede ou baixar seus e-mails. No capítulo sobre acesso remoto, veremos como é possível criar um túnel seguro entre seu micro e o gateway da rede, usando o SSH, permitindo assim encriptar todo o tráfego.

Chaves WEP Podem ser Quebradas

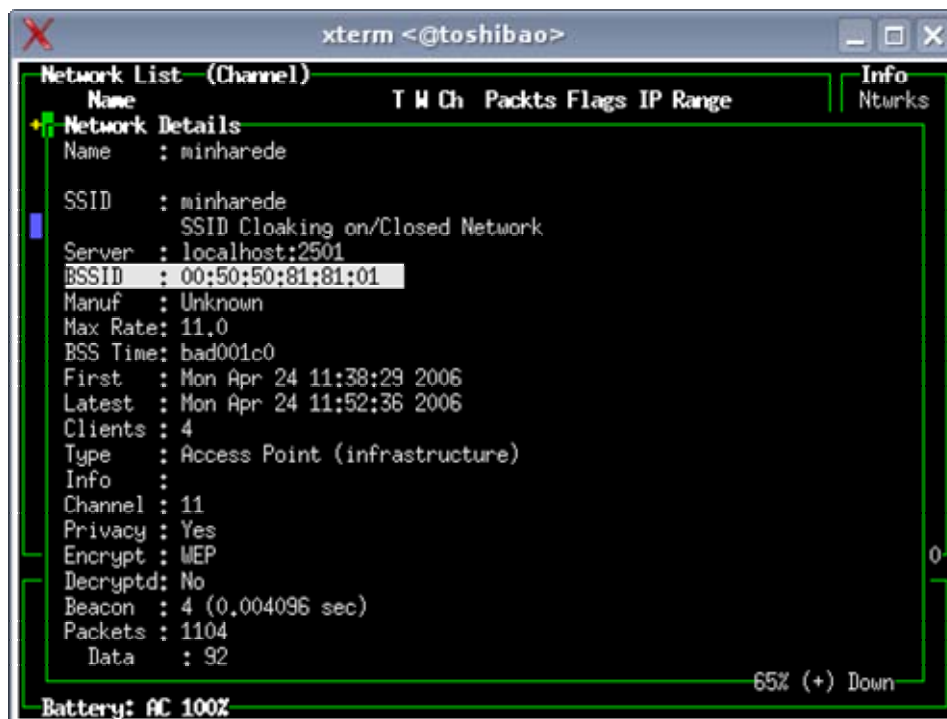
Para você entender a importância de usar o SSH o outros protocolos seguros ao usar uma rede wireless, vou falar um pouco mais sobre como quebrar chaves de encriptação, para que você possa entender os ataques usados pelos que estão do outro lado.

Alguns pontos de acesso utilizam versões vulneráveis do WEP, que são muito rápidas de quebrar (em muitos casos você pode corrigir através de uma atualização de firmware) mas, mesmo as versões "não vulneráveis" do WEP podem ser quebradas via força bruta, sempre que seja possível capturar um volume suficiente de tráfego da rede.

Você pode simular uma invasão na sua própria rede, para verificar qual seria o volume de trabalho necessário para invadí-la. Para isso, você vai precisar de pelo menos dois micros ou notebooks. Um deles vai ser usado como um cliente de rede normal e pode usar qualquer placa de rede, enquanto o segundo (que usaremos para simular o ataque) precisa ter uma placa compatível com o Kismet.

Configure o seu ponto de acesso ativando o WEP e desativando o Broadcast do SSID. Ou seja, faça uma configuração relativamente segura, mas faça de conta que esqueceu tudo :).

Comece abrindo o Kismet no notebook "invasor". Deixe que ele detecte as redes próximas, pressione "**s**" para ajustar a ordem dos nomes na lista, selecione sua rede e pressione "**shift + L**" para que ele trave a varredura na sua rede e deixe de bisbilhotar nas redes dos vizinhos.

Inicialmente, sua rede será detectada como "no ssid", já que o broadcast do SSID foi desativado no ponto de acesso. Mas, assim que qualquer micro se conecta ao ponto de acesso, o Kismet descobre o SSID correto. Pressione "**i**" para ver os detalhes da rede e anote o endereço MAC do ponto de acesso (BSSID), que precisaremos para iniciar o passo seguinte.

Agora que já sabemos o SSID e o MAC do ponto de acesso, falta quebrar o WEP. Para isso precisaremos do **Aircrack**, uma suíte de aplicativos para verificação de redes wireless, que pode ser encontrada no http://freshmeat.net/projects/aircrack/. Nos derivados do Debian, ele pode ser instalado via apt-get:

 # apt-get install aircrack

Outra opção é baixar o Back Track, um Live-CD baseado no Slax, que já vem com o Aircrack, Kismet e outras ferramentas úteis pré-instaladas.

O primeiro passo é capturar pacotes da rede usando o **airodump**. A sintaxe do comando é "airodump interface arquivo-de-log mac-do-ponto-de-acesso", como em:

 # airodump ath0 logrede 00:50:50:81:41:56

Você pode também indicar um canal (neste caso, ele escuta todas as redes que estejam transmitindo no canal indicado), como em:

 # airodump ath0 logrede 14

Isto gerará o arquivo "logrede.cap", que contém um dump de todos os pacotes capturados. Neste ponto você precisa esperar algum tempo até conseguir um volume razoável de pacotes. Para acelerar isso, faca com que o micro isca baixe alguns arquivos grandes a partir de outro micro da rede.

Abra outro terminal e use o **aircrack** para tentar quebrar a chave de encriptação. Você pode fazer isso sem interromper a captura do airodump, daí a idéia de usar dois terminais separados.

Ao usar o aircrack, é preciso especificar o comprimento da chave WEP (64 ou 128 bits) e o arquivo gerado pelo airodump, como em:

 # aircrack-n 64 logrede.cap

ou:

 # aircrack -n 128 logrede.cap

Caso o arquivo contenha pacotes destinados a mais de um ponto de acesso, ele pergunta qual verificar. No caso, indique sua rede.

O aircrack usa um ataque de força bruta para tentar descobrir a chave de encriptação da rede. A base do ataque são os IV's (vetores de inicialização), a parte de 24 bits da chave de encriptação, que é trocada periodicamente. O volume de IV's gerados varia de acordo com a rede (por isso existem redes mais vulneráveis que outras) mas, em teoria, é possível quebrar a encriptação de uma rede de 128 bits caso você consiga capturar de 500 mil a um milhão de IV's, enquanto que uma chave de 64 bits pode ser quebrada com pouco mais de 200 mil. Caso seja usada uma chave fácil de adivinhar, os números são drasticamente reduzidos, permitindo em muitos casos que a chave seja quebrada com a captura de alguns poucos milhares de IV's.

Como todo processo de força bruta, o tempo necessário é aleatório. O aircrack pode descobrir a chave correta tanto logo no início da captura, quanto só depois de capturar mais de um milhão de IV's. Por isso é interessante deixar o terminal de captura do airodump aberto e ir executando o aircrack periodicamente, até que ele descubra a chave. Quanto maior o volume de dados capturados, maior a possibilidade dele descobrir a chave.

Uma forma de aumentar a eficiência do ataque, ou seja, aumentar a chance de descobrir a chave, usando o mesmo número de IV's é aumentar o fudge factor, o que faz com que o aircrack teste um número maior de combinações. Isso naturalmente aumenta proporcionalmente o tempo necessário para o teste. O default do aircrack é 2, você pode alterar o valor usando a opção "-f", como em:

 # aircrack -f 4 -n 128 logrede.cap

É comum começar fazendo um teste com o valor default, depois com fudge 4 (como no exemplo) e a partir daí ir dobrando até descobrir a chave, ou a demora se tornar inviável.

Com um grande volume de IV's, uma chave WEP de 64 bits é um alvo fácil. Neste caso a quebra demorou apenas 21 segundos:

Como é necessário capturar um grande volume de dados e muitas redes são usadas apenas para acessar a Internet e outras tarefas leves, capturar o volume de pacotes necessário poderia demorar dias.

Um invasor com um nível mediano de conhecimento, provavelmente não se contentaria em esperar tanto tempo. Ao invés disso, ele poderia usar um ataque de flood para induzir tráfego na sua rede, de forma a acelerar o processo, transformando os muitos dias em apenas alguns minutos.

Um exemplo de ferramenta usada para este tipo de ataque é o **aireplay**, mais um integrante da equipe do aircrack. O comando abaixo lança um chopchop atack (o tipo de ataque mais eficiente para quebrar chaves WEP) contra o ponto de acesso referente ao endereço MAC especificado, através da interface ath0:

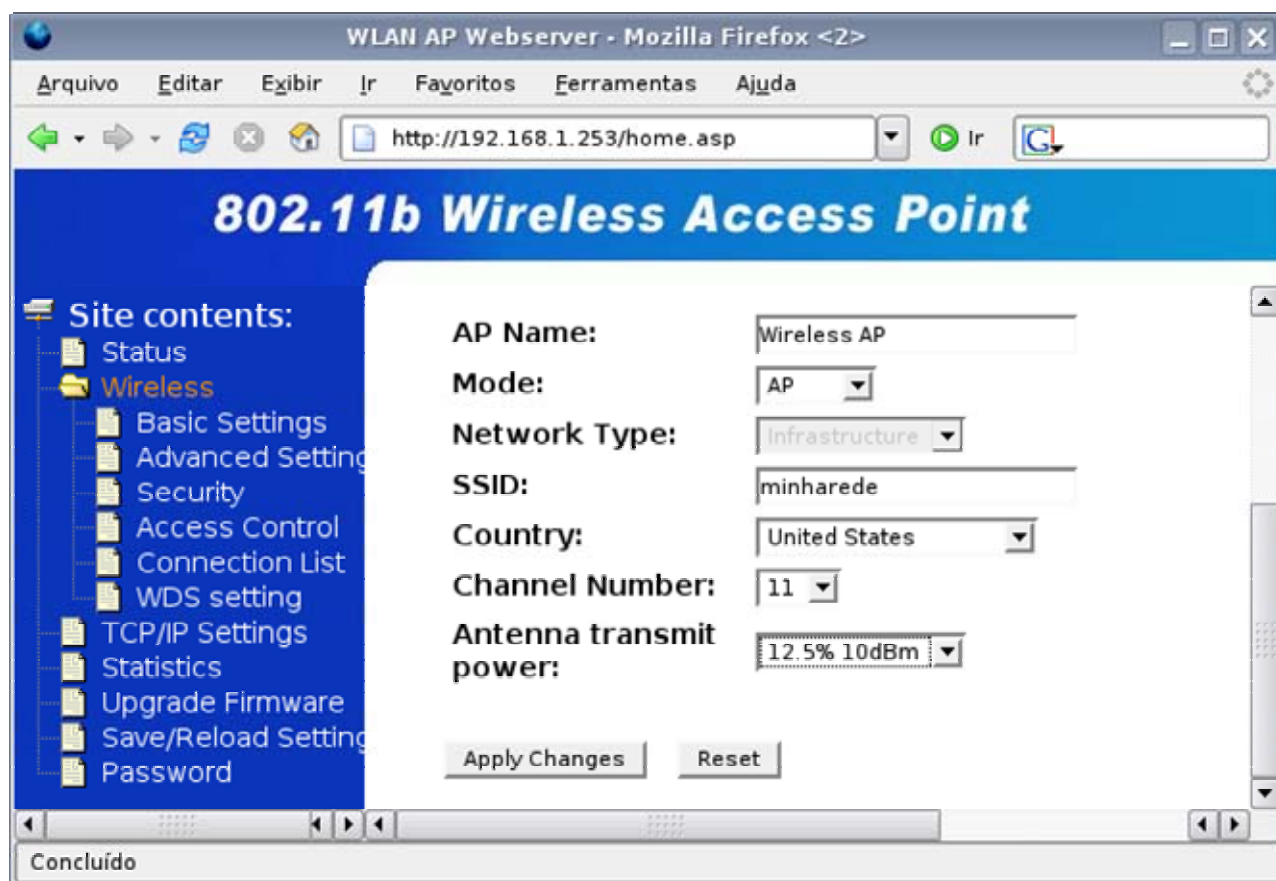
 # aireplay -b 00:50:50:81:81:01 -x 512 ath0 -4

Neste ataque, o aireplay captura um weak packet emitido por algum dos outros micros conectados ao ponto de acesso e o repete indefinidamente, obrigando o ponto de acesso a responder e assim aumentar rapidamente a contagem de IV's, permitindo quebrar a chave WEP muito mais rapidamente. Este é o tipo de ataque mais efetivo, pois derruba a última grande barreira contra a quebra do WEP, que era justamente a demora em capturar um grande volume de pacotes.

O "-x 512" especifica o número de pacotes que serão enviados por segundo. Aumentar o número permite quebrar a chave mais rapidamente, mas por outro lado vai reduzir o desempenho da rede, o que pode levar o administrador a perceber o ataque (a menos que ele seja feito em um momento de ociosidade da rede).

Como pode ver, o WEP dificulta o acesso à rede, mas quebrá-lo é apenas questão de tempo. Para melhorar a segurança da sua rede, o ideal é combinar várias camadas de segurança e monitorar os acessos, fazendo com que o tempo e trabalho necessário para invadir a rede seja maior (o que vai afastar os curiosos e invasores casuais) e vai lhe dar tempo para detectar e investigar casos mais graves. Ao usar o WEP, o ideal é trocar a chave de encriptação regularmente, de forma que, mesmo que alguém consiga descobrir a chave, não consiga usar a rede por muito tempo antes que ela seja trocada. Se possível, utilize o WPA, que, apesar dos problemas de compatibilidade, é muito mais seguro.

Caso a rede seja usada apenas dentro de um pequeno espaço, como uma única sala, apartamento ou escritório, você pode também reduzir a potência do transmissor no ponto de acesso, o que acaba sendo uma medida muito efetiva, pois realmente impede que o sinal da rede seja captado de longe.

Procure pela opção "Antenna transmit power" ou similar e veja qual é o menor valor com que a rede funciona corretamente dentro da área necessária:

Outra dica que dificulta, é habilitar a restrição de acesso à rede com base no endereço MAC, geralmente disponível através da opção "Access Control" do ponto de acesso. Ao ativar esta opção, você cria uma lista com os endereços MAC das placas autorizadas e o ponto de acesso restringe o acesso de qualquer outra.

Programas como o airodump e o próprio Kismet permitem descobrir o endereço MAC dos micros que estão acessando determinada rede muito facilmente, e o endereço MAC da placa de rede pode ser forjado (no Linux, por exemplo, você pode falsear usando o comando "ifconfig wlan0 hw ether 00:11:D8:76:59:2E", onde você substitui o "wlan0" pela interface e o "00:11:D8:76:59:2E" pelo endereço MAC desejado). A questão é que, ao forjar o endereço, o invasor vai derrubar o micro com a placa clonada, de forma que você perceba que algo está errado.

O próximo passo seria isolar sua rede wireless do restante da rede, fazendo com que, o invasor possa acessar a Internet, mas não tenha como acessar compartilhamentos e outros recursos da rede.

O mais simples neste caso é instalar uma placa de rede adicional no servidor da rede (ou em qualquer outro micro na ausência dele), onde é conectado o ponto de acesso. Compartilhe a conexão com a placa do AP, mas utilize duas faixas de IP's separados, com um firewall ativo, configurado para bloquear tentativas de conexão provenientes dos micros dentro da rede wireless.